

"last accessed" + ↑


PORTAL
 US Patent & Trademark Office
 THE ACM DIGITAL LIBRARY

[Subscribe \(Full Service\)](#) [Register \(Limited Service, Free\)](#) [Login](#)
 Search: The ACM Digital Library The Guide

Terms used probability pointer mutex

Found 21 of 148,786

Sort results by relevance
 Display results expanded form

Save results to a Binder
 Search Tips
 Open results in a new window

Try an Advanced Search
 Try this search in The ACM Guide

Results 1 - 20 of 21

Result page: 1 [2](#)Relevance scale **1 Waiting algorithms for synchronization in large-scale multiprocessors**

Beng-Hong Lim, Anant Agarwal

August 1993 **ACM Transactions on Computer Systems (TOCS)**, Volume 11 Issue 3Full text available:  pdf(2.72 MB)Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Through analysis and experiments, this paper investigates two-phase waiting algorithms to minimize the cost of waiting for synchronization in large-scale multiprocessors. In a two-phase algorithm, a thread first waits by polling a synchronization variable. If the cost of polling reaches a limit Lpoll and further waiting is necessary, the thread is blocked, incurring an additional fixed cost, B. The choice of Lpoll

Keywords: barriers, blocking, competitive analysis, locks, producer-consumer synchronization, spinning, waiting time

2 Shared memory objects: An almost non-blocking stack

Hans-J. Boehm

July 2004 **Proceedings of the twenty-third annual ACM symposium on Principles of distributed computing**Full text available:  pdf(174.83 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

Non-blocking data structure implementations can be useful for performance and fault-tolerance reasons. And they are far easier to use correctly in a signal- or interrupt-handler context. We describe a weaker class of "almost non-blocking" data structures, which block only if more than some number N of threads attempt to simultaneously access the same data structure. We argue that this gives much of the benefit of fully non-blocking data structures, particularly for signal or interrupt hand ...

Keywords: compare-and-swap, interrupt handler, linked list, lock-free, memory allocation, non-blocking, signal handler, stack

3 Implementation of resilient, atomic data types

William Weihl, Barbara Liskov

April 1985 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 7 Issue 2Full text available:  pdf(2.19 MB)Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

A major issue in many applications is how to preserve the consistency of data in the presence of concurrency and hardware failures. We suggest addressing this problem by implementing applications in terms of abstract data types with two properties: Their objects are atomic (they provide serializability and recoverability for activities using them) and

resilient (they survive hardware failures with acceptably high probability). We define what it means for abstract data types to be atomic and ...

4 Specification and implementation of resilient, atomic data types

William Weihl, Barbara Liskov

June 1983 **Proceedings of the 1983 ACM SIGPLAN symposium on Programming language issues in software systems**

Full text available:  pdf(1.28 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

A major issue in many applications is how to preserve the consistency of data in the presence of concurrency and hardware failures. We suggest addressing this problem by implementing applications in terms of abstract data types with two properties: Their objects are atomic (they provide serializability and recoverability for activities using them) and resilient (they survive hardware failures with acceptably high probability). We define what it means for abstract data types to be atomic and ...

5 Implementation of Argus

B. Liskov, D. Curtis, P. Johnson, R. Scheifler

November 1987 **ACM SIGOPS Operating Systems Review , Proceedings of the eleventh ACM Symposium on Operating systems principles**, Volume 21 Issue 5

Full text available:  pdf(1.34 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Argus is a programming language and system developed to support the construction and execution of distributed programs. This paper describes the implementation of Argus, with particular emphasis on the way we implement atomic actions, because this is where Argus differs most from other implemented systems. The paper also discusses the performance of Argus. The cost of actions is quite reasonable, indicating that action systems like Argus are practical.

6 Reliable object storage to support atomic actions

Brian M. Oki, Barbara H. Liskov, Robert W. Scheifler

December 1985 **ACM SIGOPS Operating Systems Review , Proceedings of the tenth ACM symposium on Operating systems principles**, Volume 19 Issue 5

Full text available:  pdf(939.13 KB)

Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

7 The linked class of Modula-3

E. Levy

August 1988 **ACM SIGPLAN Notices**, Volume 23 Issue 8

Full text available:  pdf(732.03 KB)

Additional Information: [full citation](#), [abstract](#), [index terms](#)

The desirability of providing built-in linked list support for the next generation of system programming languages is discussed. A Modula-2 pseudo inheritance implementation is given that demonstrates a proposed set of primitives. A set of rules are specified that enhance reliability without adding significant restrictions. Benefits to concurrency control are described.

8 Guardians and actions: linguistic support for robust, distributed programs

Barbara Liskov, Robert Scheifler

January 1982 **Proceedings of the 9th ACM SIGPLAN-SIGACT symposium on Principles of programming languages**

Full text available:  pdf(1.34 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)

This paper presents an overview of an integrated programming language and system designed to support the construction and maintenance of distributed programs: programs in which modules reside and execute at communicating, but geographically distinct, nodes. The language is intended to support a class of applications in which the manipulation and preservation of long-lived, on-line, distributed data is important. The language addresses

the writing of robust programs that survive hardware failures ...

9 Introduction to Demos

Graham Birtwistle

December 1981 **Proceedings of the 13th conference on Winter simulation - Volume 2**

Full text available: [pdf\(1.11 MB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

Demos [1,2] is yet another discrete event simulation language hosted in Simula. It was released in 1979 and is now running on IBM, DEC, UNIVAC, and CDC hardwares amongst others. The paper contains a short introduction to Simula's object and context features; an explanation of the process approach to simulation; a brief comparison of Simula and GPSS; and finally, the main features of Demos are presented via an example.

10 Session: Modula-3 network objects over ANSA: heterogeneous object-based RPC in a modern systems programming language

David Evers, Peter Robinson

September 1992 **Proceedings of the 5th workshop on ACM SIGOPS European workshop: Models and paradigms for distributed systems structuring**

Full text available: [pdf\(453.29 KB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#)

Modula-3 provides language-level features such as threads, objects and exceptions which are useful in distributed systems. The ANSA testbench provides a complete infrastructure for object-based distributed systems, but currently requires the use of C as the main programming language. We describe a successful attempt to marry the two, which provides a practical example of how a modern systems programming language can make the construction of object-based distributed systems more congenial for the ...

11 Experience Using Multiprocessor Systems—A Status Report

Anita K. Jones, Peter Schwarz

June 1980 **ACM Computing Surveys (CSUR)**, Volume 12 Issue 2

Full text available: [pdf\(4.48 MB\)](#)

Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

12 A foundation for an efficient multi-threaded scheme system

Suresh Jagannathan, Jim Philbin

January 1992 **ACM SIGPLAN Lisp Pointers , Proceedings of the 1992 ACM conference on LISP and functional programming**, Volume V Issue 1

Full text available: [pdf\(1.19 MB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

We have built a parallel dialect of Scheme called STING that differs from its contemporaries in a number of important respects. STING is intended to be used as an operating system substrate for modern parallel programming languages. The basic concurrency management objects in STING are first-class lightweight threads of control and virtual processors (VPs). Unlike high-level concurrency structures, STING threads and VPs are not encumbered by complex synchronization protocols. ...

13 Techniques for obtaining high performance in Java programs

Iffat H. Kazi, Howard H. Chen, Berdenia Stanley, David J. Lilja

September 2000 **ACM Computing Surveys (CSUR)**, Volume 32 Issue 3

Full text available: [pdf\(816.13 KB\)](#)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

This survey describes research directions in techniques to improve the performance of programs written in the Java programming language. The standard technique for Java execution is interpretation, which provides for extensive portability of programs. A Java interpreter dynamically executes Java bytecodes, which comprise the instruction set of the Java Virtual Machine (JVM). Execution time performance of Java programs can be improved

through compilation, possibly at the expense of portability ...

Keywords: Java, Java virtual machine, bytecode-to-source translators, direct compilers, dynamic compilation, interpreters, just-in-time compilers

14 E-commerce: SweetDeal: representing agent contracts with exceptions using XML rules, ontologies, and process descriptions

Benjamin N. Grosof, Terrence C. Poon

May 2003 **Proceedings of the twelfth international conference on World Wide Web**

Full text available:  pdf(481.94 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

SweetDeal is a rule-based approach to representation of business contracts that enables software agents to create, evaluate, negotiate, and execute contracts with substantial automation and modularity. It builds upon the situated courteous logic programs knowledge representation in RuleML, the emerging standard for Semantic Web XML rules. Here, we newly extend the SweetDeal approach by also incorporating process knowledge descriptions whose ontologies are represented in DAML+OIL (emerging standa ...

Keywords: DAML+OIL, OWL, RDF, XML, business process automation, declarative, description logic, electronic commerce, electronic contracts, intelligent software agents, knowledge representation, knowledge-based, logic programs, ontologies, process descriptions, process knowledge, rules, semantic web, semantic web services, web services

15 Growing languages with metamorphic syntax macros

Claus Brabrand, Michael I. Schwartzbach

January 2002 **ACM SIGPLAN Notices , Proceedings of the 2002 ACM SIGPLAN workshop on Partial evaluation and semantics-based program manipulation**, Volume 37 Issue 3

Full text available:  pdf(217.81 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)

"From now on, a main goal in designing a language should be to plan for growth." Guy Steele: *Growing a Language*, OOPSLA '98 invited talk. We present our experiences with a syntax macro language which we claim forms a general abstraction mechanism for growing (domain-specific) extensions of programming languages. Our syntax macro language is designed to guarantee type safety and termination. A concept of *metamorphisms* allows the arguments of a macro to be inductively def ...

16 EMERALDS: a small-memory real-time microkernel

Khawar M. Zuberi, Padmanabhan Pillai, Kang G. Shin

December 1999 **ACM SIGOPS Operating Systems Review , Proceedings of the seventeenth ACM symposium on Operating systems principles**, Volume 33 Issue 5

Full text available:  pdf(1.59 MB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

EMERALDS (Extensible Microkernel for Embedded, ReAL-time, Distributed Systems) is a real-time microkernel designed for small-memory embedded applications. These applications must run on slow (15-25MHz) processors with just 32-128 kbytes of memory, either to keep production costs down in mass-produced systems or to keep weight and power consumption low. To be feasible for such applications, the OS must not only be small in size (less than 20 kbytes), but also have low-overhead kernel services. Un ...

17 Network objects

Andrew Birrell, Greg Nelson, Susan Owicki, Edward Wobber

December 1993 **ACM SIGOPS Operating Systems Review , Proceedings of the fourteenth ACM symposium on Operating systems principles**, Volume 27 Issue 5

Full text available:  pdf(1.35 MB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

A network object is an object whose methods can be invoked over a network. This paper describes the design, implementation, and early experience with a network objects system for Modula-3. The system is novel for its overall simplicity. The paper includes a thorough description of realistic marshaling algorithms for network objects.

18 Analyzing communication latency using the Nectar communication processor

Peter Steenkiste

October 1992 **ACM SIGCOMM Computer Communication Review , Conference proceedings on Communications architectures & protocols**, Volume 22 Issue 4

Full text available:  pdf(1.16 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

For multicomputer applications, the most important performance parameters of a network is the latency for short messages. In this paper we present an analysis of communication latency using measurement of the Nectar system. Nectar is a high-performance multicomputer built around a high-bandwidth crosspoint network. Nodes are connected to the Nectar network using network coprocessors that are primarily responsible the protocol processing, but that can also execute application code. This arch ...

19 CPU reservations and time constraints: efficient, predictable scheduling of independent activities

Michael B. Jones, Daniela Roșu, Marcel-Cătălin Roșu

October 1997 **ACM SIGOPS Operating Systems Review , Proceedings of the sixteenth ACM symposium on Operating systems principles**, Volume 31 Issue 5

Full text available:  pdf(2.25 MB)

Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

20 Automatic performance setting for dynamic voltage scaling

Krisztián Flautner, Steve Reinhardt, Trevor Mudge

September 2002 **Wireless Networks**, Volume 8 Issue 5

Full text available:  pdf(328.69 KB)

Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

The emphasis on processors that are both low power and high performance has resulted in the incorporation of dynamic voltage scaling into processor designs. This feature allows one to make fine granularity tradeoffs between power use and performance, provided there is a mechanism in the OS to control that tradeoff. In this paper, we describe a novel software approach to automatically controlling dynamic voltage scaling in order to optimize energy use. Our mechanism is implemented in the Linux ke ...

Keywords: dynamic voltage scaling, interactive performance, performance-setting, power management, response time

Results 1 - 20 of 21

Result page: [1](#) [2](#)

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2005 ACM, Inc.

[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads:  [Adobe Acrobat](#)  [QuickTime](#)  [Windows Media Player](#)  [Real Player](#)


[Subscribe \(Full Service\)](#) [Register \(Limited Service, Free\)](#) [Login](#)
 The ACM Digital Library The Guide

THE ACM DIGITAL LIBRARY

[Feedback](#) [Report a problem](#) [Satisfaction survey](#)
Terms used **probability pointer mutex**

Found 21 of 148,786

Sort results by

[Save results to a Binder](#)
[Try an Advanced Search](#)

Display results

[Search Tips](#)
[Try this search in The ACM Guide](#)
 [Open results in a new window](#)

Results 21 - 21 of 21

Result page: [previous](#) [1](#) [2](#)

Relevance scale

**21** [Firefly: a multiprocessor workstation](#)

Charles P. Thacker, Lawrence C. Stewart

October 1987 **Proceedings of the second international conference on Architectural support for programming languages and operating systems**, Volume 15 , 22 , 21 Issue 5 , 10 , 4

Full text available: [pdf\(1.10 MB\)](#) Additional Information: [full citation](#), [abstract](#), [citations](#), [index terms](#)

Firefly is a shared-memory multiprocessor workstation that contains from one to seven MicroVAX 78032 processors, each with a floating point unit and a sixteen kilobyte cache. The caches are coherent, so that all processors see a consistent view of main memory. A system may contain from four to sixteen megabytes of storage. Input-output is done via a standard DEC QBus. Input-output devices are an Ethernet controller, fixed disks, and a monochrome 1024 x 768 display with keyboard and mouse. Option ...

Results 21 - 21 of 21

Result page: [previous](#) [1](#) [2](#)

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2005 ACM, Inc.

[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads: [Adobe Acrobat](#) [QuickTime](#) [Windows Media Player](#) [Real Player](#)

 **PORTAL**
US Patent & Trademark Office
"future access" + ↗

Subscribe (Full Service) Register (Limited Service, Free) Login
Search: The ACM Digital Library The Guide
+pointer +entry +probability

THE ACM DIGITAL LIBRARY  Feedback Report a problem Satisfaction survey

Terms used pointer entry probability

Found 14 of 227 searched out of 227.

Sort results by relevance Save results to a Binder
 Search Tips
 Display results expanded form Open results in a new window

Try an Advanced Search
 Try this search in The ACM Guide

Results 1 - 14 of 14

Relevance scale 

1 Memory system analysis and optimization: Dynamic tracking of page miss ratio curve for memory management

Pin Zhou, Vivek Pandey, Jagadeesan Sundaresan, Anand Raghuraman, Yuanyuan Zhou, Sanjeev Kumar

October 2004 **Proceedings of the 11th international conference on Architectural support for programming languages and operating systems**

Full text available:  pdf(281.38 KB) Additional Information: full citation, abstract, references, index terms

Memory can be efficiently utilized if the dynamic memory demands of applications can be determined and analyzed at run-time. The page miss ratio curve(MRC), i.e. page miss rate vs. memory size curve, is a good performance-directed metric to serve this purpose. However, dynamically tracking MRC at run time is challenging in systems with virtual memory because not every memory reference passes through the operating system (OS). This paper proposes two methods to dynamically track MRC of application ...

Keywords: memory management, power management, resource allocation

2 Implementation and performance of integrated application-controlled file caching, prefetching, and disk scheduling

Pei Cao, Edward W. Felten, Anna R. Karlin, Kai Li

November 1996 **ACM Transactions on Computer Systems (TOCS)**, Volume 14 Issue 4

Full text available:  pdf(609.00 KB) Additional Information: full citation, abstract, references, citations, index terms

As the performance gap between disks and micropocessors continues to increase, effective utilization of the file cache becomes increasingly immportant. Application-controlled file caching and prefetching can apply application-specific knowledge to improve file cache management. However, supporting application-controlled file caching and prefetching is nontrivial because caching and prefetching need to be integrated carefully, and the kernel needs to allocate cache blocks among processes ap ...

Keywords: application-controlled resource management, disk scheduling, file caching, file prefetching

3 Distance Associativity for High-Performance Energy-Efficient Non-Uniform Cache Architectures

Zeshan Chishti, Michael D. Powell, T. N. Vijaykumar

December 2003 **Proceedings of the 36th Annual IEEE/ACM International Symposium on Microarchitecture**

Full text available:  pdf(189.85 KB) Additional Information: [full citation](#), [abstract](#), [index terms](#)

 Publisher Site

Wire delays continue to grow as the dominant component of latency for large caches. A recent work proposed an adaptive, non-uniform cache architecture (NUCA) to manage large, on-chip caches. By exploiting the variation in access time across widely-spaced subarrays, NUCA allows fast access to close subarrays while retaining slow access to far subarrays. While the idea of NUCA is attractive, NUCA does not employ design choices commonly used in large caches, such as sequential tag-data access for low power. Mo ...

4 Trace-driven memory simulation: a survey

Richard A. Uhlig, Trevor N. Mudge

June 1997 **ACM Computing Surveys (CSUR)**, Volume 29 Issue 2

Full text available:  pdf(636.11 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

As the gap between processor and memory speeds continues to widen, methods for evaluating memory system designs before they are implemented in hardware are becoming increasingly important. One such method, trace-driven memory simulation, has been the subject of intense interest among researchers and has, as a result, enjoyed rapid development and substantial improvements during the past decade. This article surveys and analyzes these developments by establishing criteria for evaluating trac ...

Keywords: TLBs, caches, memory management, memory simulation, trace-driven simulation

5 A high-level abstraction of shared accesses

Peter J. Keleher

February 2000 **ACM Transactions on Computer Systems (TOCS)**, Volume 18 Issue 1

Full text available:  pdf(183.57 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#), [review](#)

We describe the design and use of the tape mechanism, a new high-level abstraction of accesses to shared data for software DSMs. Tapes consolidate and generalize a number of recent protocol optimizations, including update-based locks and recorded-replay barriers. Tapes are usually created by "recording" shared accesses. The resulting recordings can be used to anticipate future accesses by tailoring data movement to application semantics. Tapes-based mechanisms a ...

Keywords: DSM, programming libraries, shared memory, update protocols

6 A framework for efficient reuse of binary code in Java

Pramod G. Joisha, Samuel P. Midkiff, Mauricio J. Serrano, Manish Gupta

June 2001 **Proceedings of the 15th international conference on Supercomputing**

Full text available:  pdf(419.49 KB) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

This paper presents a compilation framework that enables efficient sharing of executable code across distinct Java Virtual Machine (JVM) instances. High-performance JVMs rely on run-time compilation, since static compilation cannot handle many dynamic features of Java. These JVMs suffer from large memory footprints and high startup costs, which are serious problems for embedded devices (such as hand held personal digital assistants and cellular phones) and scalable servers. A recently propose ...

7 Revisitation patterns in World Wide Web navigation

Linda Tauscher, Saul Greenberg

March 1997 **Proceedings of the SIGCHI conference on Human factors in computing systems**

Full text available:  pdf(984.35 KB) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

Keywords: WWW, history mechanisms, hypertext, navigation, web

8 Flexible use of memory for replication/migration in cache-coherent DSM multiprocessors

Vijayaraghavan Soundararajan, Mark Heinrich, Ben Verghese, Kourosh Gharachorloo, Anoop Gupta, John Hennessy

April 1998 **ACM SIGARCH Computer Architecture News , Proceedings of the 25th annual international symposium on Computer architecture**, Volume 26 Issue 3

Full text available:  [pdf\(1.76 MB\)](#)  Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)
[Publisher Site](#)

Given the limitations of bus-based multiprocessors, CC-NUMA is the scalable architecture of choice for shared-memory machines. The most important characteristic of the CC-NUMA architecture is that the latency to access data on a remote node is considerably larger than the latency to access local memory. On such machines, good data locality can reduce memory stall time and is therefore a critical factor in application performance. In this paper we study the various options available to system desi ...

9 Hive: fault containment for shared-memory multiprocessors

J. Chapin, M. Rosenblum, S. Devine, T. Lahiri, D. Teodosiu, A. Gupta

December 1995 **ACM SIGOPS Operating Systems Review , Proceedings of the fifteenth ACM symposium on Operating systems principles**, Volume 29 Issue 5

Full text available:  [pdf\(1.90 MB\)](#) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

10 Contention elimination by replication of sequential sections in distributed shared memory programs

Honghui Lu, Alan L. Cox, Willy Zwaenepoel

June 2001 **ACM SIGPLAN Notices , Proceedings of the eighth ACM SIGPLAN symposium on Principles and practices of parallel programming**, Volume 36 Issue 7

Full text available:  [pdf\(173.49 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

In shared memory programs contention often occurs at the transition between a sequential and a parallel section of the code. As all threads start executing the parallel section, they often access data just modified by the thread that executed the sequential section, causing a flurry of data requests to converge on that processor. We address this problem in a software distributed shared memory system by replicating the execution of the sequential sections on all pr ...

11 Combined performance gains of simple cache protocol extensions

F. Dahlgren, M. Dubois, P. Stenström

April 1994 **ACM SIGARCH Computer Architecture News , Proceedings of the 21ST annual international symposium on Computer architecture**, Volume 22 Issue 2

Full text available:  [pdf\(1.22 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

We consider three simple extensions to directory-based cache coherence protocols in shared-memory multiprocessors. These extensions are aimed at reducing the penalties associated with memory accesses and include a hardware prefetching scheme, a migratory sharing optimization, and a competitive-update mechanism. Since they target different components of the read and write penalties, they can be combined effectively. Detailed architectural simulations using five benchmarks show substantial combined ...

12 An integrated memory management scheme for dynamic alias resolution

Tzi-cker Chiueh

August 1991 **Proceedings of the 1991 ACM/IEEE conference on Supercomputing**Full text available:  pdf(944.90 KB) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)**13 Translating collocations for bilingual lexicons: a statistical approach**

Frank Smadja, Kathleen R. McKeown, Vasileios Hatzivassiloglou

March 1996 **C mputational Linguistics**, Volume 22 Issue 1Full text available:   Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)
[Publisher Site](#)

Collocations are notoriously difficult for non-native speakers to translate, primarily because they are opaque and cannot be translated on a word-by-word basis. We describe a program named Champollion which, given a pair of parallel corpora in two different languages and a list of collocations in one of them, automatically produces their translations. Our goal is to provide a tool for compiling bilingual lexical information above the word level in multiple languages, for different domains. The a ...

**14 Compiler-based I/O prefetching for out-of-core applications**

Angela Demke Brown, Todd C. Mowry, Orran Krieger

May 2001 **ACM Transactions on Computer Systems (TOCS)**, Volume 19 Issue 2Full text available:  Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#), [review](#)

Current operating systems offer poor performance when a numeric application's working set does not fit in main memory. As a result, programmers who wish to solve "out-of-core" problems efficiently are typically faced with the onerous task of rewriting an application to use explicit I/O operations (e.g., read/write). In this paper, we propose and evaluate a fully automatic technique which liberates the programmer from this task, provides high performance, and requires only minima ...

Keywords: compiler optimization, prefetching, virtual memory

Results 1 - 14 of 14

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2005 ACM, Inc.

[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads:  [Adobe Acrobat](#)  [QuickTime](#)  [Windows Media Player](#)  [Real Player](#)

Welcome to IEEE Xplore®

- [Home](#)
- [What Can I Access?](#)
- [Log-out](#)

Tables of Contents

- [Journals & Magazines](#)
- [Conference Proceedings](#)
- [Standards](#)

Search

- [By Author](#)
- [Basic](#)
- [Advanced](#)
- [CrossRef](#)

Member Services

- [Join IEEE](#)
- [Establish IEEE Web Account](#)
- [Access the IEEE Member Digital Library](#)

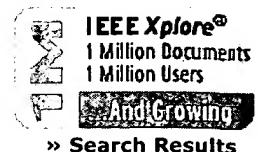
IEEE Enterprise

- [Access the IEEE Enterprise File Cabinet](#)

 [Print Format](#)



RELEASE 1.8

Welcome
United States Patent and Trademark Office

Help FAQ Terms IEEE Peer Review

Quick Links

Welcome to IEEE Xplore®

- Home
- What Can I Access?
- Log-out

Tables of Contents

- Journals & Magazines
- Conference Proceedings
- Standards

Search

- By Author
- Basic
- Advanced
- CrossRef

Member Services

- Join IEEE
- Establish IEEE Web Account
- Access the IEEE Member Digital Library

IEEE Enterprise

- Access the IEEE Enterprise File Cabinet

 Print FormatYour search matched **5** of **1112315** documents.A maximum of **500** results are displayed, **15** to a page, sorted by **Relevance in Descending** order.

Refine This Search:

You may refine your search by editing the current search expression or entering a new one in the text box.

access probability and future Check to search within this result set

Results Key:

JNL = Journal or Magazine **CNF** = Conference **STD** = Standard**1 An adaptive network prefetch scheme***Jiang, Z.; Kleinrock, L.;*

Selected Areas in Communications, IEEE Journal on , Volume: 16 , Issue: 3 , April 1998

Pages:358 - 368

[Abstract] [PDF Full-Text (340 KB)] IEEE JNL

2 Call admission control using dynamic cell virtual traffic in CDMA systems*El-Said, M.M.; Kumar, A.; Bennett, E.E.; Foley, A.N.; Hillenbrand, T.J.;*

Vehicular Technology Conference, 2003. VTC 2003-Fall. 2003 IEEE 58th , Volume: 5 , 6-9 Oct. 2003

Pages:3508 - 3511 Vol.5

[Abstract] [PDF Full-Text (318 KB)] IEEE CNF

3 Measurement-based admission control in UMTS multiple cell case*Elayoubi, S.-E.; Chahed, T.; Hebuterne, G.;*

Personal, Indoor and Mobile Radio Communications, 2003. PIMRC 2003. 14th IEEE Proceedings on , Volume: 2 , 7-10 Sept. 2003

Pages:1770 - 1774 vol.2

[Abstract] [PDF Full-Text (407 KB)] IEEE CNF

4 A new prefetch cache scheme*Shun-Zheng Yu; Kobayashi, H.;*

Global Telecommunications Conference, 2000. GLOBECOM '00. IEEE , Volume: 1 , 27 Nov.-1 Dec. 2000

Pages:350 - 355 vol.1

[Abstract] [PDF Full-Text (588 KB)] IEEE CNF

5 Performance evaluation of a modified PRMA-HS protocol*Wen Pingping; Gu Xuemai;*

Communications, Circuits and Systems and West Sino Expositions, IEEE 2002
International Conference on , Volume: 1 , 29 June-1 July 2002
Pages:450 - 454 vol.1

[\[Abstract\]](#) [\[PDF Full-Text \(297 KB\)\]](#) [IEEE CNF](#)

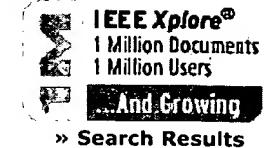
[Home](#) | [Log-out](#) | [Journals](#) | [Conference Proceedings](#) | [Standards](#) | [Search by Author](#) | [Basic Search](#) | [Advanced Search](#) | [Join IEEE](#) | [Web Account](#) | [New this week](#) | [OPAC](#)
[Linking Information](#) | [Your Feedback](#) | [Technical Support](#) | [Email Alerting](#) | [No Robots Please](#) | [Release Notes](#) | [IEEE Online Publications](#) | [Help](#) | [FAQ](#) | [Terms](#) | [Back to Top](#)

Copyright © 2004 IEEE — All rights reserved


[Membership](#) [Publications/Services](#) [Standards](#) [Conferences](#) [Careers/Jobs](#)


RELEASE 1.8

 Welcome
 United States Patent and Trademark Office

[Help](#) [FAQ](#) [Terms](#) [IEEE Peer Review](#)
Quick Links


» Search Results

Welcome to IEEE Xplore®

- Home
- What Can I Access?
- Log-out

Tables of Contents

- Journals & Magazines
- Conference Proceedings
- Standards

Search

- By Author
- Basic
- Advanced
- CrossRef

Member Services

- Join IEEE
- Establish IEEE Web Account
- Access the IEEE Member Digital Library

IEEE Enterprise

- Access the IEEE Enterprise File Cabinet

Print Format

[Home](#) | [Log-out](#) | [Journals](#) | [Conference Proceedings](#) | [Standards](#) | [Search by Author](#) | [Basic Search](#) | [Advanced Search](#) | [Join IEEE](#) | [Web Account](#) | [New this week](#) | [OPAC](#)
[Linking Information](#) | [Your Feedback](#) | [Technical Support](#) | [Email Alerting](#) | [No Robots Please](#) | [Release Notes](#) | [IEEE Online Publications](#) | [Help](#) | [FAQ](#) | [Terms](#) | [Back to Top](#)

Copyright © 2004 IEEE — All rights reserved



US 20020112116A1

(19) United States

(12) Patent Application Publication
Nelson(10) Pub. No.: US 2002/0112116 A1
(43) Pub. Date: Aug. 15, 2002

(54) METHODS, SYSTEMS, AND COMPUTER PROGRAM PRODUCTS FOR STORING DATA IN COLLECTIONS OF TAGGED DATA PIECES

(52) U.S. Cl. 711/103; 711/156; 711/170

(76) Inventor: Mark Edward Nelson, Holly Springs, NC (US)

(57) ABSTRACT

52 bits

Correspondence Address:
MYERS BIGEL SIBLEY & SAJOVEC
PO BOX 37428
RALEIGH, NC 27627 (US)

to recent

(21) Appl. No.: 10/086,218

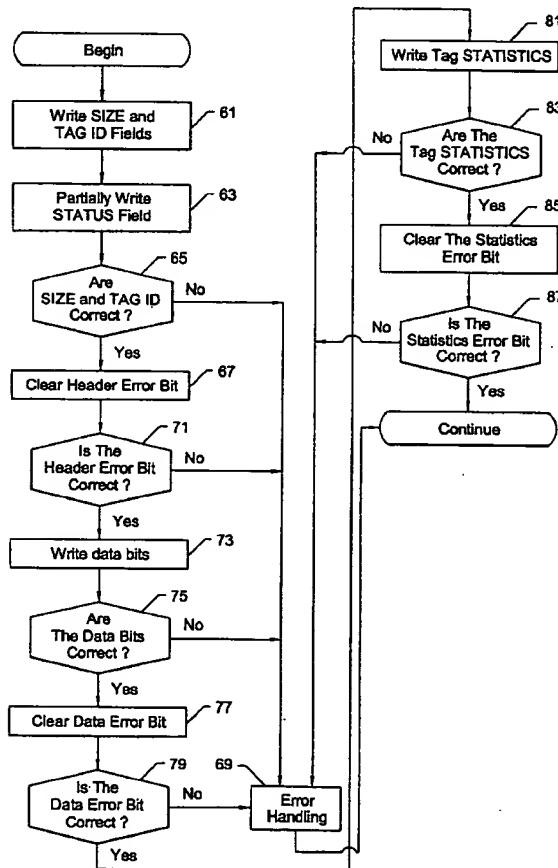
(22) Filed: Feb. 26, 2002

Related U.S. Application Data

(63) Continuation-in-part of application No. 09/715,337, filed on Nov. 17, 2000.

Methods of storing collections of related data in memory can include storing a first collection of related data in a first collection of tagged data pieces and storing a second collection of related data in a second collection of tagged data pieces. Each tagged data piece of the first collection can include a respective header including a first tag identification common to each of the tagged data pieces of the first collection and a fragment identification unique to each of the tagged data pieces of the first collection. Similarly, each tagged data piece of the second collection can include a respective header including a second tag identification common to each of the tagged data pieces of the second collection and a fragment identification unique to each of the tagged data pieces of the second collection wherein the first and second tag identifications are distinct. Related systems and computer program products are also discussed.

Publication Classification

(51) Int. Cl.⁷ G06F 12/00

PGPUB-DOCUMENT-NUMBER: 20020112116

PGPUB-FILING-TYPE: new

DOCUMENT-IDENTIFIER: US 20020112116 A1

TITLE: Methods, systems, and computer program products for storing data in collections of tagged data pieces

PUBLICATION-DATE: August 15, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Nelson, Mark Edward	Holly Springs	NC	US	

US-CL-CURRENT: 711/103, 711/156 , 711/170

ABSTRACT:

Methods of storing collections of related data in memory can include storing a first collection of related data in a first collection of tagged data pieces and storing a second collection of related data in a second collection of tagged data pieces. Each tagged data piece of the first collection can include a respective header including a first tag identification common to each of the tagged data pieces of the first collection and a fragment identification unique to each of the tagged data pieces of the first collection. Similarly, each tagged data piece of the second collection can include a respective header including a second tag identification common to each of the tagged data pieces of the second collection and a fragment identification unique to each of the tagged data pieces of the second collection wherein the first and second tag identifications are distinct. Related systems and computer program products are also discussed.

----- KWIC -----

Detail Description Paragraph - DETX (82):

[0108] When a tagged data piece of a collection with a corresponding fragment list saved, is relocated as a result of sector reclamation, or is updated with new data, the new memory device address for the tagged data piece is entered into the fragment list (in place of the old memory device address for the fragment) in fragment identification order. The fragment list can be maintained in dynamic memory of the memory controller 33 as long as the data collection corresponding to the fragment list is being used by the processor 35. Once the data collection is no longer being used or is not expected to be used in the near future, the fragment list for the data collection can be de-allocated or removed from dynamic memory 34.

Detail Description Paragraph - DETX (84):

[0110] A pointer for the all list 331 of FIG. 20 can also be maintained to identify the last fragment list accessed by the memory controller. If the memory controller has last accessed data collection B, the pointer can be used to identify the location of FragBLst in the all list. If the next request for a data collection is for data collection B, the memory controller 33 can quickly locate the fragment list for data collection B using the pointer and the all list 331. Otherwise, the memory controller can search the all list for the tag identification for the data collection to be accessed. If a fragment list for the data collection is currently open, the search of the all list by tag identification will allow the memory controller to quickly locate the dynamic memory address of the corresponding fragment list. If a fragment list for the data collection is not currently open, the tag identification will not be located in the all list. Accordingly, the memory controller can get the address for the sorted list from the all list, locate the tag identification in the sorted list, and then generate a fragment list for the desired data collection.

Detail Description Paragraph - DETX (85):

[0111] In other words, the pointer and the all list can be used to quickly locate open fragment lists for previously compiled data collections and/or to generate a fragment list if a fragment list is not currently opened. When data of a collection is to be retrieved, the memory controller can first check the last accessed tag identification designated by the all list pointer. Because a data collection may be accessed repeatedly, there is a relatively high probability that the last accessed data collection may be the same as the data

collection to be accessed. Accordingly, access time may be saved by first checking the last accessed data collection. If the tag identification of the last accessed data collection matches the tag identification of the data collection to be accessed, the memory controller can use the corresponding dynamic memory address to locate the appropriate fragment list.

Detail Description Paragraph - DETX (90) :

[0116] A collection of tagged data pieces can be located according to embodiments of the present invention using sorted and all lists as discussed below with reference to FIG. 30. If access to a collection is requested at block 801, the memory controller can first check the all list pointer at block 803, and if the all list pointer identifies an all list entry corresponding to the requested collection at block 805, the data can be retrieved at block 821. If the all list pointer does not identify an all list entry corresponding to the requested collection at block 805, the all list can be scanned at block 807. If an entry in the all list corresponds to the requested collection at block 809, the all list pointer can be set to the entry corresponding to the requested collection at block 819, and the data can be retrieved at block 821. If the all list does not include an entry for the requested collection at block 809, the sorted list can be scanned at block 811. If an entry in the sorted list corresponds to the requested collection at block 813, a fragment list for the requested collection can be generated at block 815, an entry for the requested collection can be added to the all list at block 817, the all list pointer can be set to the new all list entry at block 819, and the data can be retrieved at block 821. Because a collection may be accessed repeatedly, access times can be reduced by first checking the all list pointer that identifies the last accessed collection and then scanning the all list that identifies a plurality of the most recently accessed collections before scanning the sorted list of all collections.

Detail Description Paragraph - DETX (129) :

[0155] Like an index, a record set can include a list of record numbers for a table in order of a sorting criteria without including actual record data. In addition, a record set may be assembled manually by inserting specific record numbers in a specific order. A record set may also be created as a snapshot of an index. Unlike an index, a record set is stored in non-volatile memory for future use.